REMARKS

Petition for Extension of Time Under 37 CFR 1.136(a)

It is hereby requested that the term to respond to the Examiner's Action of February 22, 2010 be extended one month, from May 22, 2010 to June 22, 2010.

The Commissioner is hereby authorized to charge the extension fee and any additional fees associated with this communication to Deposit Account No. 50-4364.

In the Office Action, the Office indicated that claims 1 and 4-10 are pending in the application and the Office rejected all of the claims.

Rejections under 35 U.S.C. §103

On page 2 of the Office Action, the Office rejected claims 1, 4, 5, and 8-10 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 7,100,172 to Voellm in view of U.S. Patent No. 7,047,537 to Stern On page 6 of the Office Action, the Office rejected claims 6-7 under 35 U.S.C. §103(a) as being unpatentable over Voellum in view of Stern, and further in view of U.S. Patent No. 5,974,470 to Hammond.

The Present Invention

The claimed invention provides a more efficient way of implementing a remapping DLL, as disclosed, *inter alia*, beginning on page 13, line 12 of the application as filed. Lines 15 to 17 of page 13 state specifically that a significant feature of the remapping DLL of the claimed invention is the use of the relocation instructions in the DLL *to modify the export data table of*

the DLL itself. Claim 1 includes a requirement that the relocation instruction inserts into the export data table the address location for the function in the additional DLL (i.e., it modifies the export data table of the DLL itself). As stated in paragraph 2 of page 13, this novel use of the relocation instructions is particularly beneficial because the executable is able to refer directly to the DLL implementing the functionality required by the executable without the use of a sub routine within the remapping DLL, as is done in the prior art.

U.S. Patent No. 7,100,172 to Voellm

U.S. Patent No. 7,100,172 to Voellm ("Voellm") discloses a system and method for altering the operation of a computer application while avoiding recompiling the computer application or modifying the kernel associated with the operating system of a computing device. A computer application is launched in a suspended mode. An asynchronous procedure call (APC) is used to load *an additional* dynamic link library (DLL) to be associated with the computer application. The additional DLL includes routines that operate differently than routines originally associated with the computer application through an initial DLL. The references to the routines within the computer application are redirected to the routines of the additional (DLL). The operation of the computer application is therefore changed while avoiding rewriting the application or changing the operating system. Of significance herein, the export data table of the DLL itself is not modified.

U.S. Patent No. 7,047,537 to Stern

U.S. Patent No. 7,047,537 to Stern ("Stern") discloses a method and apparatus for linking a set of code modules for execution. A determination is made as to which code modules are going to be executed, and a hierarchical order is ascertained with regard to their execution. A "chain" is formed between the code modules such that when the first code module in the chain is executed, the remaining code modules will be executed in their hierarchical order. Stern's discussion of linking DLLs in an order for execution is an example of the prior art described in the present application at page 11, line 31 to page 13, line 10.

U.S. Patent No. 5,974,470 to Hammond

U.S. Patent No. 5,974,470 to Hammond ("Hammond") discloses a system for managing DLL modules and providing administrators of Windows-based PC's with more control over Windows modules. The Office relies on Hammond for an alleged teaching of the linking of an application program to a dynamic link library by ordinal number.

A *Prima Facie* Case of Obviousness Has Not Been Established

KSR (*KSR International Co. v. Teleflex Inc.,* 127 S. Ct. 1727, 82 USPQ2d 1385 (2007) requires that the Office provide "some articulated reasoning with some rationale underpinning to support the legal conclusion of obviousness." Further, the Office must "identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does," In addition, the Office must make "explicit" this rationale of "the apparent reason to combine the known elements in the fashion claimed," including a detailed explanation of "the effects of demands known to the design community or

present in the marketplace" and "the background knowledge possessed by a person having ordinary skill in the art."

An objective technical problem addressed by the claimed invention is referred to, *inter alia*, on page 5 of the original disclosure; namely the difficulties associated with co-coordinating DLL entry points as a system evolves from one release to the next in a manner that efficiently provides an interface of an original DLL in terms of one or more DLLs which together re-implement the functionality of the original DLL. This problem is addressed through the use of a novel form of remapping DLL.

DLLs are known and widely used in this art. It is also known that DLLs have an export data table that lists the addresses within the DLL of the functions that can be exported from the DLL, known as exported functions. In a modern computing device, when an executable is made ready for execution on the device, a loader of the operating system loads the executable from non-execute-in-place memory into execute-in-place memory. The loader also loads any DLLs required by that executable to function correctly. The DLLs contain coded instructions which are known as relocation instructions, and in known forms of DLLs prior to this invention, these relocation instructions describe how to modify a location within the executable in order to prepare the executable for execution at a specific address within the computing device memory space. Page 8 of the original disclosure describes how the address locations of the functions in DLLs are inserted into an executable being loaded. When it is necessary to revise a DLL, such as for example when it is required to support multiple sets of APIs, a remapping DLL has been used, and a typical example of such a known remapping DLL is described on page 11, line 31, to page 13, line 10, of the original disclosure.

As noted above, the claimed invention, as recited in independent claim 1 (and thus in all the claims), provides a more efficient way of implementing a remapping DLL. The claimed invention uses the relocation instructions in the DLL to *modify the export data table of the DLL itself.* The relocation instruction inserts into the export data table the address location for the function in the additional DLL. This novel, non-obvious, and specifically-claimed use of the relocation instructions is particularly beneficial because the executable is able to refer directly to the DLL implementing the functionality required by the executable, without the use of a sub routine within the remapping DLL as is done in the prior art.

None of the art cited by the Office, taken alone or in combination, teaches or suggests these claimed elements. In Voellm, a computer application is launched in a suspended mode. An asynchronous procedure call (APC) is used to load an additional dynamic link library (DLL) to be associated with the computer application. The additional DLL includes routines that operate differently than routines originally associated with the computer application through an initial DLL. Nothing in Voellm gives any teaching or suggestion of using the relocation instructions in the DLL to *modify the export data* table of the DLL itself, so that the relocation instruction inserts into the export data table the address location for the function in the additional DLL.

Stern describes the linking of DLLs for execution, using the above-mentioned "chain" technique. While interesting, the disclosure of Stern, like the disclosure of Voellm, contains neither a teaching nor a reasonable suggestion of using the relocation instructions in the DLL to modify the export data table of the DLL as is claimed herein.

Hammond is relied upon for an alleged teaching of the linking of an application program to a dynamic link library by ordinal number. As with Voellm and Stern, Hammond

contains no teaching or suggestion of providing a remapping DLL in place of the original existing DLL, and furthermore, of a remapping DLL having relocation instructions as now provided by present claims of this invention, and without any change to the loader.

In summary, the claims, as amended, are neither taught nor suggested by Voellm, Stern, and/or Hammond, whether taken alone or in combination. Accordingly, the Office is respectfully requested to reconsider and withdraw the rejection of claims 1, and 4-10 under 35 USC §103.

Conclusion

The present invention is not taught or suggested by the prior art. Accordingly, the Office is respectfully requested to reconsider and withdraw the rejection of the claims. An early Notice of Allowance is earnestly solicited.

The Commissioner is hereby authorized to charge the extension fee and any additional fees associated with this communication to applicant's Deposit Account No. 50-4364.

Respectfully submitted

June 22, 2010
Date

/Mark D. Simpson/
Mark D. Simpson, Esquire
Registration No. 32,942

SAUL EWING LLP
Centre Square West
1500 Market Street, 38th Floor
Philadelphia, PA 19102-2189
Telephone: 215 972 7880
Facsimile: 215 972 4169
Email: MSimpson@saul.com